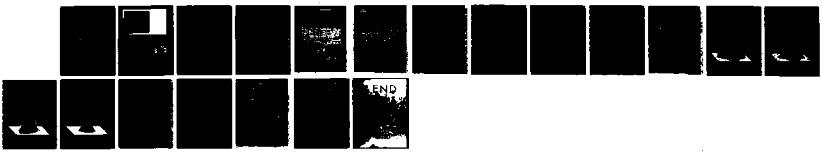
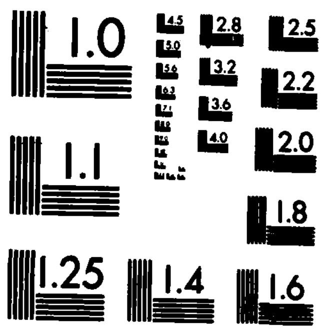
AD-A147 386 THE STABLE EVALUATION OF MULTIVARIATE B-SPLINES(U) 1/1
WISCONSIN UNIV-MADISON MATHEMATICS RESEARCH CENTER
T A GRANDINE SEP 84 MRC-TSR-2744 DAAG29-80-C-0041

UNCLASSIFIED F/G 12/1 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3

MRC Technical Summary Report # 2744

THE STABLE EVALUATION OF MULTIVARIATE B-SPLINES

Thomas A. Grandine

Mathematics Research Center
University of Wisconsin—Madison
610 Walnut Street
Madison, Wisconsin 53705

September 1984

(Received August 24, 1984)

Approved for public release
Distribution unlimited

Sponsored by

U. S. Army Research Office
P. O. Box 12211
Research Triangle Park
North Carolina 27709

UNIVERSITY OF WISCONSIN - MADISON
MATHEMATICS RESEARCH CENTER

THE STABLE EVALUATION OF MULTIVARIATE B-SPLINES

Thomas A. Grandine

Technical Summary Report #2744
September 1984

(cont fr p B)

ABSTRACT

This paper gives a general method for the stable evaluation of multivariate B-splines. The problem of evaluation along mesh boundaries is discussed in detail. Several examples are presented to demonstrate the effectiveness of the method for arbitrary B-splines. Originator-supplied Keywords include:

AMS(MOS) Subject Classification: 41A15, 65D07

Key Words: B-spline, simplex spline, ~~multivariate,~~ recurrence relation, linear programming, and simplex method.

Work Unit Number 3 - Numerical Analysis and Scientific Computing

Sponsored by the United States Army under Contract No. DAAG29-80-C-0041.

SIGNIFICANCE AND EXPLANATION

For one variable, the problem of stably evaluating B-splines via their recurrence relations is well understood. For multivariate B-splines, however, the geometry becomes more complex, and it becomes quite difficult to implement the recurrence relations in such a way that one ends up with a robust evaluation method. This paper presents a method which guarantees the stable evaluation of all smooth multivariate simplex B-splines.

to p A

The responsibility for the wording and views expressed in this descriptive summary lies with MRC, and not with the author of this report.

B

[illegible]

[illegible] A. [illegible]

In de Boor [1976], the multivariate B-spline $M(x|t_0,\dots,t_n)$ was defined to be

$$M(x|t_0,\dots,t_n) := \frac{\mathrm{vol}_{n-m}(P^{-1}x)\cap[t_0,\dots,t_n]}{\mathrm{vol}_n[t_0,\dots,t_n]}, \qquad x \in \mathbb{R}^m. \tag{1}$$

In this definition, $t_0,\dots,t_n$ are points in $\mathbb{R}^n$, $[A]$ is the convex hull of A, and P is the canonical projector

$$P : \mathbb{R}^n \to \mathbb{R}^m : x \mapsto (x(i))_1^m.$$

In addition, $\mathrm{vol}_k(A)$ is the k-dimensional volume of the set A.

This B-spline is, in general, a non-negative piecewise polynomial function of degree $\le n-m$, supported in $[Pt_0,\dots,Pt_n]$, and it is in C^{n-m-1} provided that the points $Pt_0,\dots,Pt_n$ [illegible] position [illegible] i.e. they are affinely independent. For a proof of this result, [illegible]

To evaluate the [illegible] which coincides with the spline on a given piece. The problem [illegible] begin with, the number of pieces can get to be very large [illegible] a non-trivial matter to determine in which piece a given point [illegible] the many different polynomials to evaluate cannot [illegible]

Example 1: Consider the points $t_0 = ($[illegible]$)$, $t_1 = ($[illegible]$)$, $t_2 = ($[illegible]$)$, $t_3 = (-1,0,1,0,0)$, $t_4 = (-1,-1,0,1,0)$, $t_5 = (0,-1,0,0,1)$. Let P be the canonical projector from $\mathbb{R}^5 \to \mathbb{R}^2$. Then the bivariate B-spline given by $t_0,\dots,t_5$ will be supported in $[(1,0),(1,1),$ $(0,1),(-1,0),(-1,-1),(0,-1)]$ and will be given by the [illegible] cubic pieces in the region below. It is clear that, given an arbitrary (x,y) in the plane, determining in which region (x,y) lies, without taking into account the special geometry of the situation, is a nearly hopeless task.

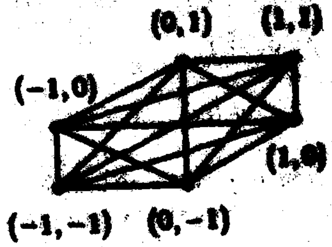


Figure 1.

A better approach involves the use of recurrence relations. In [1980], C. A. Micchelli proposed a more convenient definition of a multivariate B-spline as the distribution on $C_0^\infty(\mathbb{R}^m)$ given by

$$M(\cdot|t_0,\dots,t_n) : f \mapsto n! \int_{[t_0,\dots,t_n]} f \circ P. \tag{2}$$

Sponsored by the United States Army under Contract No. DAAG29-80-C-0041.

This definition makes sense even if $t_0, \ldots, t_n$ are not in general position. Using this definition, Micchelli was able to prove recurrence relations for these multivariate B-splines. These relations are given in Theorem 1.

Theorem 1: If $x = \sum_{i=0}^{n} \alpha_i P t_i$, with $\sum_{i=0}^{n} \alpha_i = 1$, then

$$M(x|t_0, \ldots, t_n) = \frac{n}{n-m} \sum_{i=0}^{n} \alpha_i M(x|t_0, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n). \tag{3}$$

Furthermore, if $z = \sum_{i=0}^{n} \alpha_i P t_i$, with $\sum_{i=0}^{n} \alpha_i = 0$, then

$$D_z M(\cdot|t_0, \ldots, t_n) = n \sum_{i=0}^{n} \alpha_i M(\cdot|t_0, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n). \tag{4}$$

Proof: The proof of this theorem has been given in various forms by a number of people, including Micchelli [1979], Höllig [1980], Hakopian [1980], and de Boor and Höllig [1982].

Given Theorem 1, it should now be possible to evaluate a spline. In this theorem, all the splines appearing on the right hand side of an equality are of order one less than those appearing on the left hand side. Assuming one can evaluate a piecewise constant spline in $\mathbb{R}^m$ (which amounts to computing the m-dimensional volume of a simplex), one can inductively write any spline as a linear combination of piecewise constant splines.

This is, for a specific collection of points $t_0, \ldots, t_n$, a straightforward task. The difficulty arises when one wishes to write a computer code to evaluate B-splines using (3) for *arbitrary* collections of points. The problem, of course, is to find the α_i so that $x = \sum_{i=0}^{n} \alpha_i P t_i$ and $\sum_{i=0}^{n} \alpha_i = 1$ are satisfied in the general case, in such a way that neither accuracy nor efficiency suffer. Accuracy becomes an issue whenever one is forced to use α_i of opposite signs. Since $\sum_{i=0}^{n} \alpha_i = 1$, this means that one should insist on having $\alpha_i \geq 0$, for all $i = 0, \ldots n$. Efficiency dictates that as many of the α_i as possible be zero. Thus, one wishes to find a solution to the following problem:

$$\begin{aligned} \sum_{i=0}^{n} \alpha_i P t_i &= x \\ \sum_{i=0}^{n} \alpha_i &= 1 \\ \alpha_i \geq 0 \quad &, \quad i = 0, \ldots, n \end{aligned} \tag{5}$$

The difficulty with this lies in the non-negativity constraints. Efficiency demands that at most $m+1$ of the α_i be non-zero, yet arbitrarily setting the remaining $n-m+1$ of the α_i to zero provides no guaranty that the non-negativity constraints will be satisfied.

Example 2: Taking again the bivariate B-spline given by the points $t_0, \ldots, t_5$ given in Example 1, and the point $(x, y) = (\frac{1}{2}, \frac{1}{4})$, one might wish to compute $M((x,y)|t_0, \ldots, t_5)$. Setting $\alpha_3 = \alpha_4 = \alpha_5 = 0$, one gets

$$\begin{aligned} \alpha_0 &= \frac{3}{4} \\ \alpha_1 &= -\frac{1}{4} \\ \alpha_2 &= \frac{1}{2}, \end{aligned}$$

which is not a solution to (5).

Fortunately, there is a well-established procedure for handling problems of a similar nature. Except for the absense of an objective function, (5) is of the same form as a **linear programming problem**. Such problems have been studied in great detail over the years, and many ways of computing their solutions are known. Thus, an approach which one might consider is the introduction of a "dummy" objective function to convert (5) to a linear programming problem which may then be solved by standard techniques.

In practice, this seems to work quite well. In the numerical experiments performed to date, the simplex method, implemented with the help of the **Tucker tableau** (described in the Appendix), has performed admirably. One introduces the objective function 0 (for reasons to be made clear shortly) and treats (5) as a linear programming problem; i.e., one considers the following problem, equivalent to (5):

$$\begin{aligned} & \max \quad 0 \\ & \text{subject to} \quad \sum_{i=0}^{n} \alpha_i Pt_i = x \\ & \qquad\qquad \sum_{i=0}^{n} \alpha_i = 1 \\ & \qquad\qquad \alpha_i \geq 0 \quad , \quad i = 0, \ldots, n \end{aligned} \tag{6}$$

Since the objective function is constant, solving this problem amounts to finding a feasible point. In other words, it amounts to finding a solution to (5).

In solving (6) via the method outlined in the preceding paragraph, one can see why the particular choice of an objective function is, in some sense, optimal. In the Tucker tableau, the pivot rules are such that a row of all zeros will never change. Thus, the physical storage of the row corresponding to the objective function in memory is unnecessary. Furthermore, the dual of the problem is given by

$$\begin{aligned} & \min \quad \langle x, u \rangle + v \\ & \text{subject to} \quad \langle Pt_i, u \rangle + v \geq 0 \quad , \quad i = 0, \ldots, n. \end{aligned}$$

This problem is dual feasible in that $u = 0$, $v = 0$ satisfies the constraints for this problem. This situation arises because the objective function of the primal problem (6) has non-positive coefficients. This means that (6) is "easy" to solve because one can dispense with phase I of the simplex method and use instead the **dual simplex method**.

Example 4: Consider again Example 2. Setting up this problem as a linear programming problem leads to the tableau

$$\begin{array}{c} \\ r_0 \\ r_1 \\ r_2 \end{array} \begin{array}{c} \begin{array}{ccccccc} \alpha_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & 1 \end{array} \\ \begin{pmatrix} -1 & -1 & 0 & 1 & 1 & 0 & -\frac{1}{2} \\ 0 & -1 & -1 & 0 & 1 & 1 & -\frac{1}{4} \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{pmatrix} \end{array},$$

where the variables r_0, r_1, and r_2 are so-called "slack" variables. Since (6) is made up of equality constraints, one first pivots r_0, r_1, and r_2 to the top of the tableau and, once this is done, deletes the columns corresponding to them. After all, the variables along the top are assumed to be

zero, and the deletion of a column corresponding to such a variable merely makes this condition permanent. Thus, one first exchanges r_0 and α_0 to get

$$\begin{array}{c c} & \begin{array}{ccccccc} r_0 & \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & 1 \end{array} \\ \begin{array}{c} \alpha_0 \\ r_1 \\ r_2 \end{array} & \begin{pmatrix} -1 & 1 & 0 & -1 & -1 & 0 & \frac{1}{2} \\ 0 & -1 & -1 & 0 & 1 & 1 & -\frac{1}{4} \\ -1 & 0 & -1 & -2 & -2 & -1 & -\frac{1}{2} \end{pmatrix} \end{array}.$$

Now one deletes the first column and exchanges r_1 and α_1 to get

$$\begin{array}{c c} & \begin{array}{cccccc} r_1 & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & 1 \end{array} \\ \begin{array}{c} \alpha_0 \\ \alpha_1 \\ r_2 \end{array} & \begin{pmatrix} 1 & -1 & -1 & 0 & 1 & \frac{1}{4} \\ -1 & 1 & 0 & -1 & -1 & \frac{1}{4} \\ 0 & -1 & -2 & -2 & -1 & -\frac{1}{2} \end{pmatrix} \end{array}.$$

Lastly, one deletes the column corresponding to r_1, exchanges r_2 and α_2, and deletes the resulting column corresponding to r_2 to obtain

$$\begin{array}{c c} & \begin{array}{cccc} \alpha_3 & \alpha_4 & \alpha_5 & 1 \end{array} \\ \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_2 \end{array} & \begin{pmatrix} 1 & 2 & 2 & \frac{3}{4} \\ -2 & -3 & -2 & -\frac{1}{4} \\ 2 & 2 & 1 & \frac{1}{2} \end{pmatrix} \end{array}.$$

This is not (dual) optimal, so one must exchange α_1 with some column whose entry in α_1's row is negative. Since all columns satisfy this, the first is chosen, and α_1 and α_3 are exchanged to reveal

$$\begin{array}{c c} & \begin{array}{cccc} \alpha_1 & \alpha_4 & \alpha_5 & 1 \end{array} \\ \begin{array}{c} \alpha_0 \\ \alpha_3 \\ \alpha_2 \end{array} & \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 1 & \frac{5}{8} \\ -\frac{1}{2} & \frac{3}{2} & 1 & \frac{1}{8} \\ 1 & -1 & -1 & \frac{1}{4} \end{pmatrix} \end{array}.$$

This is a solution to (6), and therefore to (5). Note that although not explicitly required, only 3, or $m+1$ of the α_i are non-zero. This occurs because (6) only had 3 constraints (other than the non-negativity constraints), and solutions of a linear programming problem must satisfy a complementarity condition; that is, the only variables which can be non-zero are those corresponding to tight constraints. Since r_0, r_1, and r_2 were forced to be zero, there can be at most 3 non-zero α_i. Thus, the linear programming approach implicitly takes care of the efficiency issue discussed above.

Once one has solved (6) by this approach, one can easily evaluate the spline using (4), assuming that the values of the lower order splines which occur on the right hand side of the equality are known. In general, this is not the case, but one can reapply the technique to each of the splines appearing on the right hand side of (4). This process may be carried out inductively until one can finally express the value of the desired B-spline at the desired point in terms of piecewise constant functions at that point.

The Tucker tableau makes this inductive process extremely efficient. The tableau which solves (6) may be used to solve the resulting subproblems. One can view each of the subproblems as being just (5) with the additional constraint $\alpha_i = 0$. Thus, to solve a subproblem, one can take

the tableau which solves the main problem, pivot α_i to the top of the tableau, delete the column corresponding to it, and then optimize. This will yield a solution to the subproblem in short order, often only one pivot. These solution tableaus for the subproblems may then be used to obtain cheap solutions to the sub-subproblems, etc. This exploitation of similarity among the various linear programming problems which one must solve saves an enormous amount of work.

At first glance, it would seem that the difficulties in computing the value of a multivariate B-spline have been overcome. Unfortunately, one of the more persistent of the problems has yet to be overcome. The piecewise constant functions which one ultimately ends up with have discontinuities along certain boundaries, namely along the **grid lines**. A grid line is a set consisting of the convex hull of m or fewer points taken from the set $\{Pt_0, ..., Pt_n\}$. A point x is said to lie on a grid line if it is a member of some such set. Whenever one wishes to evaluate a spline at a point lying on such a grid line, one runs the risk of computing it improperly.

Example 3: Suppose one wishes to evaluate $M(0,0|t_0,t_1,t_2,t_3)$, where $t_0 = (1,1,0)$, $t_1 = (-1,1,0)$, $t_2 = (-1,-1,0)$, and $t_3 = (1,-1,1)$. Then $(0,0) = \frac{1}{2}Pt_0 + \frac{1}{2}Pt_2$, and therefore

$$M(0,0|t_0,t_1,t_2,t_3) = \frac{3}{2}M(0,0|t_1,t_2,t_3) + \frac{3}{2}M(0,0|t_0,t_1,t_3).$$

But $M(\bullet|t_1,t_2,t_3)$ and $M(\bullet|t_0,t_1,t_3)$ are discontinuous at $(0,0)$, so it is unclear whether to choose the interior or exterior limits as values for these splines. If interior limits are chosen, the computed value of the spline will be twice as great as the actual value. If exterior limits are chosen, the value will be zero, and this is obviously also incorrect.

There are many ways of attempting to circumvent this problem, but nearly all fail in some way, especially when one takes into account the inexact nature of the arithmetic performed by the computer. In general, there seems to be no reasonable way to handle this problem, but for smooth splines, there are a few things one might try.

The obvious approach is to prohibit one from evaluating a spline on the grid lines. This is certainly the most sure-fire answer, and it is also a simple enough scheme to be easily implemented. All one need do whenever one finds that he is on a grid line is to move the point in some direction by ε and try again. Higher order splines are, in general, continuous, so this small change in the location of the point will make a very small change in the value of the spline. Unfortunately, this must be done by hand, since the computer is unable to tell when a point is actually "on" a grid line; the best it can usually do is to tell when a point is "near" a grid line. As the number of variables increases, the structure of the grid lines becomes increasingly complex. As a result, it becomes increasingly difficult to avoid computing the value of the spline there. For example, in one variable, the grid lines consist only of the **knots**, while in two variables, the grid lines consist of the knots as well as the line segments joining the knots (see Figure 1).

In one variable, the problem isn't so terrible. Typically, one decides that all the piecewise constant splines are either continuous from the left or continuous from the right. Then, when one needs to evaluate at a knot, one sets the value of the spline to zero if it is the left knot, for example, and non-zero if it is the right knot. Thus, for one variable at least, little needs to be done to clear up this nuisance.

Conveniently, such a plan of attack generalizes to more than one variable. One merely chooses (somewhat arbitrarily) some direction in $\mathbb{R}^m$ and evaluates piecewise constant splines according to the following rule: If x is in the interior of the region of support, or if x is on the boundary of the region of support and the arbitrarily chosen direction points into the interior, the value of the piecewise constant spline shall be used; in all other cases the value of the spline shall be 0. In theory, this rule eliminates the difficulty. Unfortunately, because of roundoff error, one can have both situations occurring, and the ambiguity about what to do persists. Thus, for

multivariate splines an alternate approach must be taken. Its successful implementation depends on the following theorem.

Theorem 2: Let $t_0, ..., t_{n+1}$ be a collection of points in general position in $\mathbf{R}^n$. Let $A := [t_0, ..., t_{n+1}]$, and let $A_i := [t_0, ..., t_{i-1}, t_{i+1}, ..., t_{n+1}]$. Then for all $x \in A$, with x not on any of the grid lines, x lies in exactly two of the A_i.

Proof: The statement $x \in A$ is equivalent to the statement that there exists a solution to the problem:

$$\begin{aligned} \sum_{i=0}^{n+1} \alpha_i t_i &= x \\ \sum_{i=0}^{n+1} \alpha_i &= 1 \\ \alpha_i \geq 0, &\qquad i = 0, ...n+1. \end{aligned} \tag{7}$$

Ignoring for the moment the inequality constraint, one can view (7) as a linear system, namely

$$T\alpha = \beta,$$

where T is an $n+1 \times n+2$ matrix, α is the vector whose individual components are the α_i, and β is the vector obtained by adding the component 1 to the end of x. T clearly has rank $n+1$, since the t_i are in general position. Thus, this linear system has a one-parameter family of solutions, say $\alpha(s) := y + sz$, where $y, z \in \mathbf{R}^{n+1}$ and $s \in \mathbf{R}$. Now one can consider the inequality constraints, $\alpha_i \geq 0$, $i = 0, ..., n+1$. This is equivalent to $y_i + sz_i \geq 0$, $i = 0, ..., n+1$. Taken together, all these conditions define some interval $S := [s_-, s_+]$ in which s must lie in order for $\alpha(s)$ to satisfy the inequality constraints. Since $x \in A$, it is clear that S is non-empty. Furthermore, it is clear that $\sum_{i=0}^{n+1} y_i = 1$ and $\sum_{i=0}^{n+1} z_i = 0$, for $\sum_{i=0}^{n+1} \alpha_i = 1$, independent of s. Since the solution cannot be unique, at least one of the z_i is non-zero. But $\sum_{i=0}^{n+1} z_i = 0$, so there must be at least two of the z_i non-zero and of opposite sign. When z_i is positive, one gets a lower bound for s, while z_i negative gives an upper bound for s. Hence, S is a finite interval. Since $\alpha(s)$ is a continuous function of s, no components of α can have sign changes in S. Furthermore, s outside of S means that one or more components of $\alpha(s)$ are negative there, hence must change sign on the boundary of S. Suppose $\alpha_i(s_-) = 0$ and $\alpha_j(s_+) = 0$. Since x does not lie on a grid line, $\alpha_i(s_-)$ is the only zero component of α at s_-. Similarly, $\alpha_j(s_+)$ is the only zero component of α at s_+. But this says that $x \in A_i$ and $x \in A_j$. Since $\alpha(s)$ is affine, no other solutions with a zero component are possible. Thus, x lies in exactly two of the A_i. This proves the theorem.

With this theorem, one can now correctly evaluate continuous piecewise linear B-splines. Suppose one wishes to evaluate $M(x|t_0, ..., t_{n+1})$, where $Pt_0, ..., Pt_{n+1}$ are points in general position in $\mathbf{R}^n$. Then, after one solves (5) to get

$$M(x|t_0, ..., t_{n+1}) = (n+1) \sum_{i=0}^{n+1} \alpha_i M(x|t_0, ..., t_{i-1}, t_{i+1}, ..., t_{n+1}),$$

it is clear that if x does not lie on a grid line, then all but one of the $M(x|t_0, ..., t_{i-1}, t_{i+1}, ..., t_{n+1})$ will be zero. This is an immediate consequence of Theorem 2. If x is on a grid line, however, one can still impose this condition. This trick forces evaluation on the grid lines to behave just like evaluation off the grid lines. One must be careful, however, to choose the correct piecewise constant spline to be non-zero.

Example 5: Consider again Example 4. One imposes the condition that exactly one of $M(\bullet|t_1,t_2,t_3)$ and $M(\bullet|t_0,t_2,t_3)$ can be non-zero at $(0,0)$. In this case, it doesn't matter what one chooses. However, one might attempt to solve this problem numerically with the following result:

$$M(0,0|t_0,t_1,t_2,t_3) = \frac{3}{2}M(0,0|t_1,t_2,t_3) + 10^{-6}M(0,0|t_0,t_2,t_3) + \frac{3}{2}M(0,0|t_0,t_1,t_3).$$

One clearly cannot choose $M(\bullet|t_0,t_2,t_3)$ as the only non-zero spline and expect to get reasonable results.

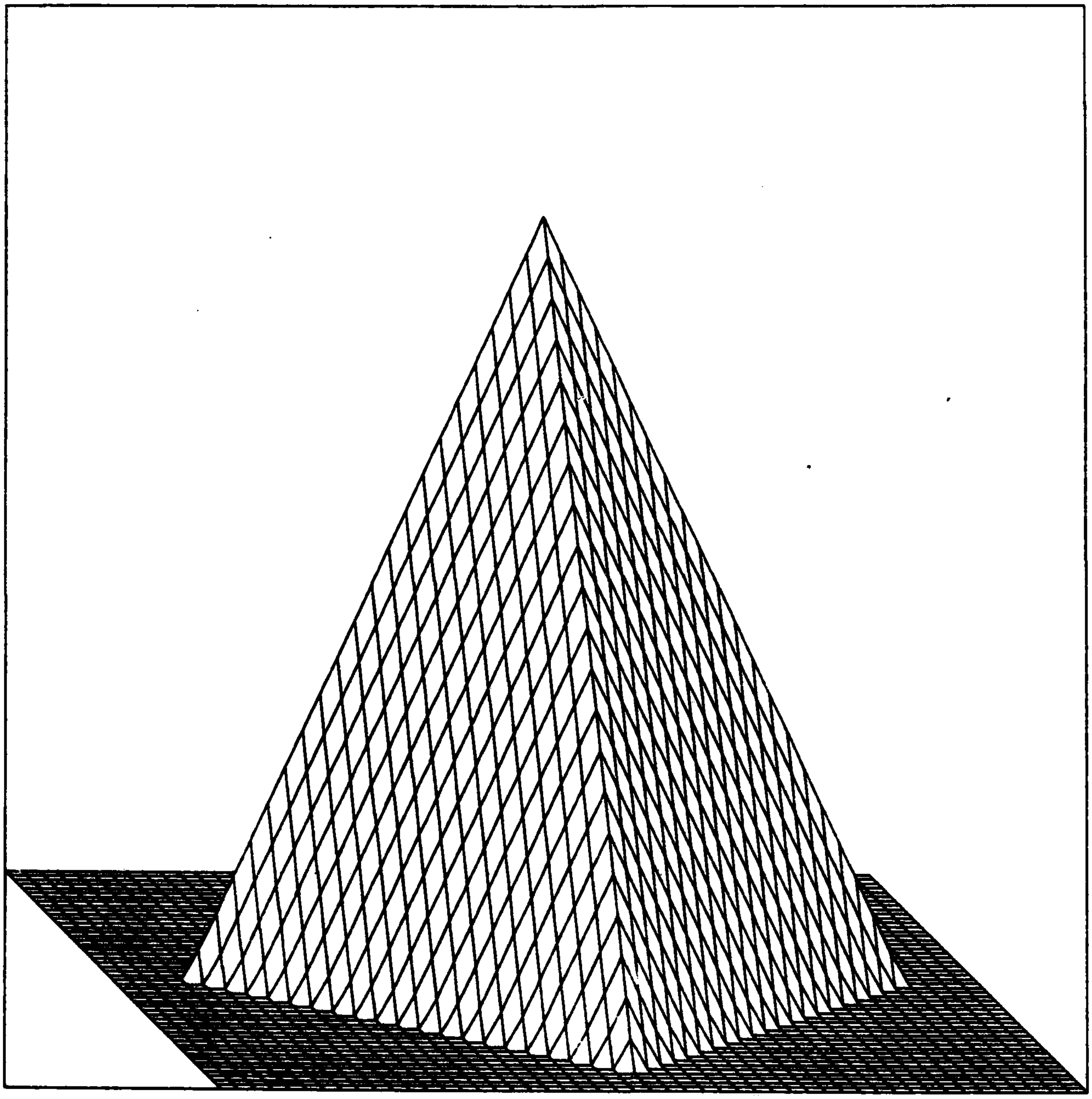
This indicates that one must be careful in the selection of the non-zero spline. A good method is to choose from all possible splines the one which has the largest coefficient. This will eliminate the kind of numeric nuisance which occurs in Example 5.

However, it must be pointed out that this approach only works if the linear spline is continuous. If it is discontinuous, one can evaluate the spline stably everywhere except along the discontinuity, where numeric noise makes the exact location of the discontinuity impossible to calculate. Fortunately, for smooth splines at least, this never happens, and one needn't be concerned with it.

Given that one can evaluate continuous linear splines stably everywhere, one can evaluate smooth higher degree splines stably everywhere. Instead of expressing the higher degree spline as a linear combination of constant splines, one stops one level sooner and expresses it as a linear combination of linear splines, each of which is continuous and can be evaluated stably by the method just described. One no longer worries about extraneous terms resulting from grid line effects because the linear splines are continuous.

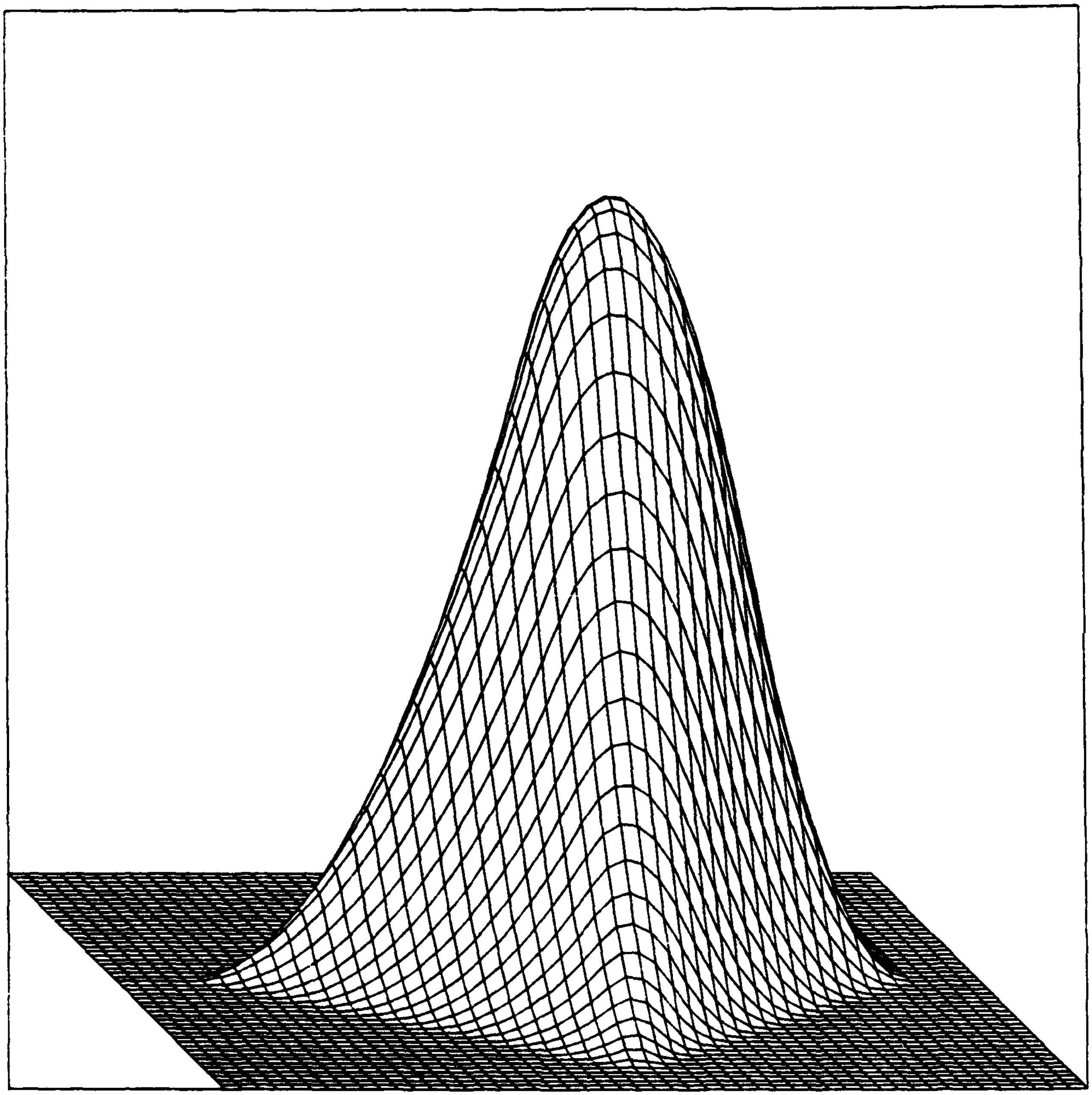
Using this technique, many different splines have been computed in the bivariate case. Some of these appear on the following pages. In order to produce these graphs, the mesh was deliberately chosen so that many evaluations along grid lines were necessary. The knots used for each spline are given at the bottom of each page.

Linear Bivariate B−spline



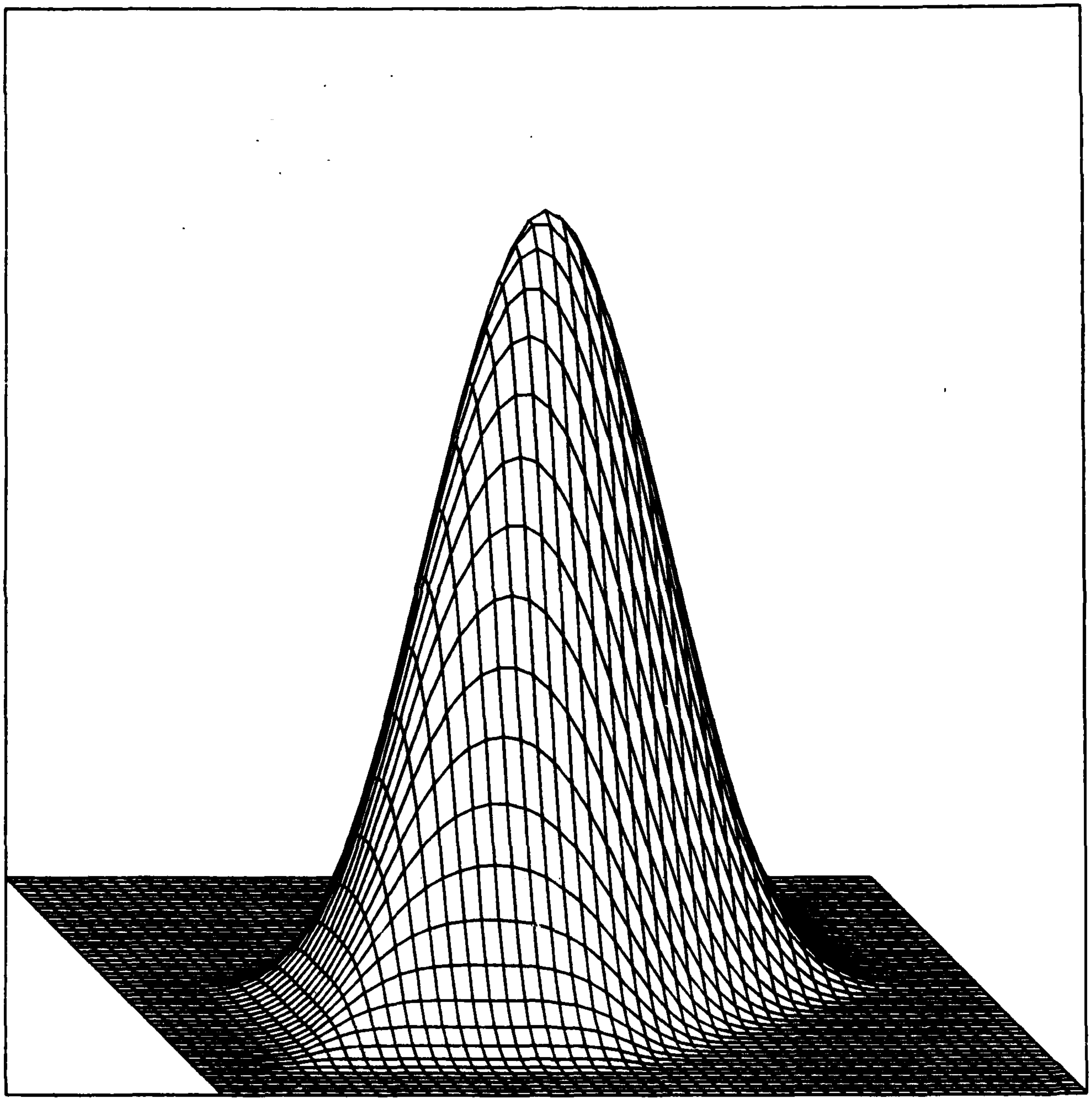
(1,0),(0,1),(−1,0),(0,−1)

Quadratic Bivariate B-spline



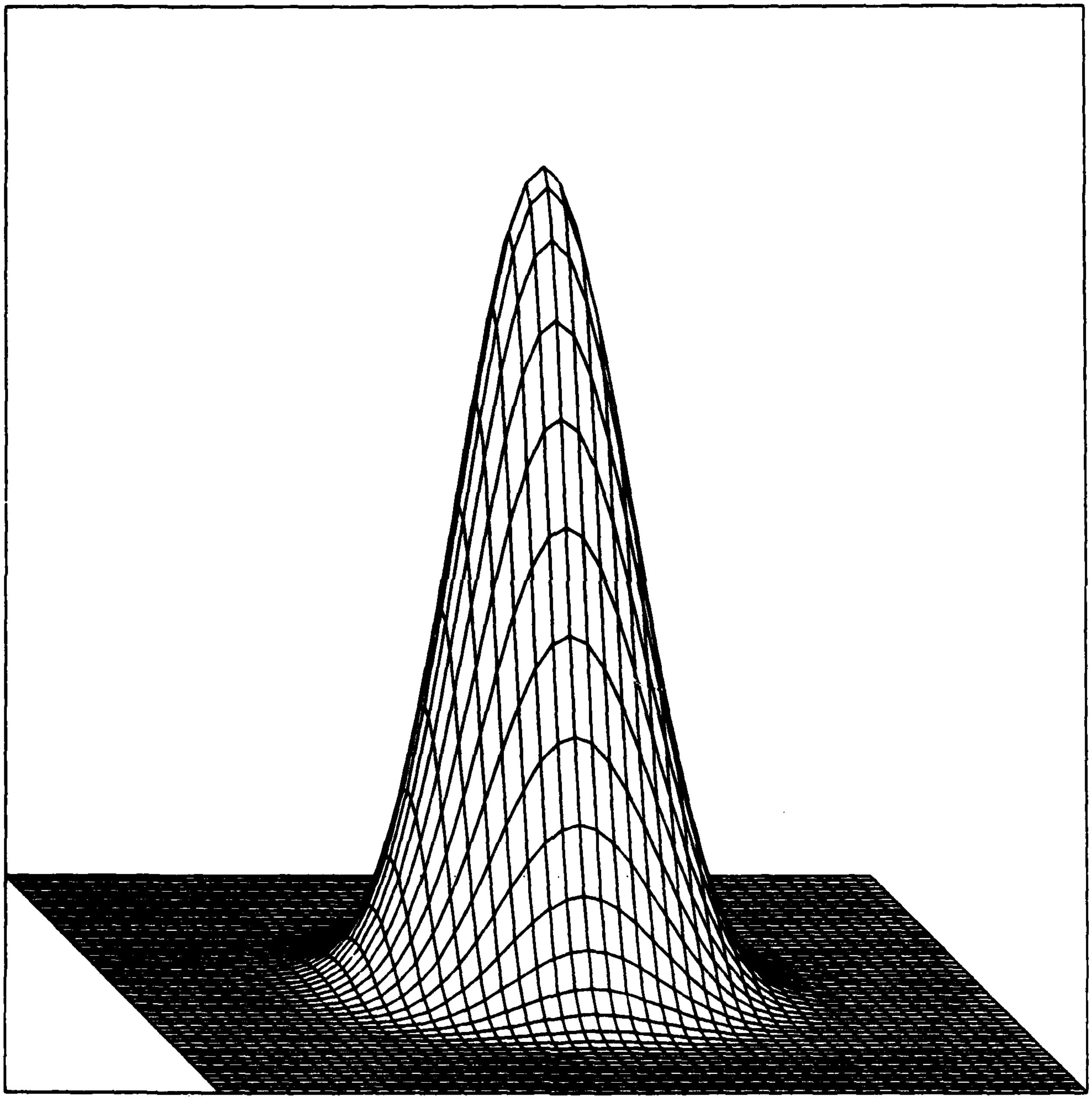
(1,0),(0,1),(−1,0),(0,−1),(1,1)

Cubic Bivariate B–spline



(1,0),(1,1),(0,1),(−1,0),(−1,−1),(0,−1)

Degree 7 Bivariate B-spline



$(\sin 2\pi k/10, \cos 2\pi k/10),\ k=1,\ldots,10$

Appendix: The Tucker Tableau

Consider the following linear programming problem:

$$\begin{array}{rl} \min & z := \langle c, x \rangle - d \\ \text{subject to} & r := b - Ax \geq 0 \\ & x_i \geq 0, \quad i = 1, \ldots, n. \end{array} \tag{A1}$$

Each constraint, $r_i \geq 0$ or $x_i \geq 0$, defines a half-space in which the solution must lie. Since one must satisfy all the constraints, the solution must lie in the intersection of these various half-spaces, i.e., x must lie in some polyhedral region in $\mathbb{R}^n$, the **feasible region**. The functional which one wishes to minimize over this region is linear, so the solution must lie at an extreme point of the feasible region, sometimes called a **vertex**. A vertex is, in general, determined by specifying n of the bounding hyperplanes to which it belongs. This means that a vertex is determined by setting n of the numbers $r_1, r_2, \ldots, r_m, x_1, x_2, \ldots, x_n$ to zero. These are the **non-basic variables** for that vertex; call them $\xi_1, \xi_2, \ldots, \xi_n$. The remaining variables are the basic variables for that vertex, and they will be denoted by $\rho_1, \rho_2, \ldots, \rho_m$. They are related by the equation

$$\rho = \beta - T\xi, \tag{A2}$$

which is obtained from the equation

$$r = b - Ax \tag{A3}$$

by partial Gauss-Jordan elimination, i.e., by solving (A3) for $\rho_1, \rho_2, \ldots, \rho_m$. Initially, $\xi = x$ and $\rho = r$, i.e., one is at the vertex $x = 0$. The vertex specified is feasible exactly when $\beta \geq 0$ (since that makes all the basic variables non-negative while the non-basic ones are zero by choice). In order to keep track of the value of the **objective function**, z, one expresses it always in terms of the non-basic variables,

$$z = \langle \gamma, \xi \rangle - \delta. \tag{A4}$$

Initially, $\gamma = c$ and $\delta = d$. For the computations, one keeps the essential information in the **Tucker tableau**, as follows:

$$\begin{array}{c} \\ \rho_1 \\ \rho_2 \\ \vdots \\ \rho_m \\ z \end{array} \begin{array}{c} \begin{array}{ccccc} -\xi_1 & -\xi_2 & \cdots & -\xi_n & 1 \end{array} \\ \left(\begin{array}{ccccc} t_{11} & t_{12} & \cdots & t_{1n} & \beta_1 \\ t_{21} & t_{22} & \cdots & t_{2n} & \beta_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ t_{m1} & t_{m2} & \cdots & t_{mn} & \beta_m \\ \gamma_1 & \gamma_2 & \cdots & \gamma_n & \delta \end{array} \right) \end{array}. \tag{A5}$$

The simplex method operates on this tableau in the following way: If each entry in the final column is non-negative, then one is currently located at a feasible vertex. One wishes to move to a neighboring vertex where the value of the objective function will be less. If one of the entries in the final row is negative, then one can increase the value of the corresponding non-basic variable and reduce the size of the objective function. This is done by exchanging the non-basic variable with a basic variable. This amounts to solving the equation representing the basic variable for the non-basic variable and then substituting the resulting expression into all of the other equations.

Suppose one wishes to exchange the variable in the k-th row with the variable in the l-th column. The following **pivot rules** can be derived to carry out this process. Let T represent the

entire tableau, and t_{ij} its i,j entry. Then

$$\begin{aligned} &t_{kl} \to 1/t_{kl} \\ &t_{il} \to t_{il}/t_{kl} \qquad\qquad\quad i = 1,\ldots,k-1,k+1,\ldots,m+1 \\ &t_{kj} \to -t_{kj}/t_{kl} \quad , \qquad j = 1,\ldots,l-1,l+1,\ldots,n+1 \; . \\ &t_{ij} \to t_{ij} - t_{il}t_{kj}/t_{kl} \end{aligned} \tag{A6}$$

In order to maintain feasibility, one must choose the pivot row carefully. This is done by selecting the row k by the following rule:

$$\text{Choose } k \text{ so that } \quad \frac{t_{k,n+1}}{t_{kl}} = \min_i \left\{ \frac{t_{k,n+1}}{t_{il}} \middle| t_{il} > 0 \right\}. \tag{A7}$$

If more than one row satisfies this, then any of them may be chosen. One performs exchanges until the final row of the tableau is non-negative. One then has an optimal solution.

Of course, if the rightmost column has some negative entries, then there is some extra work involved before one can start this optimization process; one must first find a feasible vertex. Fortunately, the early pioneers of linear programming noted that the simplex method can even be used to tackle this problem. After all, it is a procedure which transforms a tableau into one whose last row is non-negative, while preserving the non-negativity of the left column. So, one simply considers the dual simplex method, in which the roles of basic and non-basic variables are reversed. A new objective function (0 works well because it eliminates the necessity of adding a row to the tableau) is added to the problem. This objective function should have the property that its coefficients in terms of the current non-basic variables are non-negative, but no other restrictions need to be placed on its selection. Now one has a tableau whose last row is non-negative, but whose last column does not share this property. One can apply the dual simplex method to this tableau to get one in which both the last row and the last column are non-negative. The dual simplex method proceeds just as the simplex method proceeds, except that the row and column pivot rules are interchanged, the former pivot column selection rule is now used to choose the pivot row, and the column l is chosen by the following rule:

$$\text{Choose } l \text{ so that } \quad -\frac{t_{m+1,l}}{t_{kl}} = \min_j \left\{ -\frac{t_{m+1,l}}{t_{kj}} \middle| t_{kj} < 0 \right\}. \tag{A8}$$

One continues exchanging basic and non-basic variables until the rightmost column is non-negative. At this point, one has a feasible vertex, and the new objective function can be replaced by the original, which, of course, must be expressed in terms of the current non-basic variables, something best accomplished by keeping track of it all along. One can now proceed with the optimization process as before.

This entire procedure, as well as many of its alternatives, is described in Dantzig [1963]. Mangasarian [1978] discusses the Tucker tableau and many of its applications.

References

[1] de Boor, Carl [1976]. "Splines as linear combinations of B-splines," in *Approximation Theory II*, G. G. Lorentz, C. K. Chui, and L. L. Schumaker, eds., Academic Press, pp. 1–47.

[2] de Boor, Carl [1982]. "Topics in Multivariate Approximation Theory," in *Topics in Numerical Analysis*, P. Turner, ed. *Springer Lecture Notes in Mathematics* **965**, 1982, pp. 39–78.

[3] de Boor, Carl, and Höllig, Klaus [1981]. "Recurrence Relations for Multivariates B-splines," Mathematics Research Center TSR #2215. *Proc. Amer. Math. Soc.*, to appear.

[4] de Boor, Carl, and Höllig, Klaus [1982]. "B-splines from Parallelepipeds," Mathematics Research Center TSR #2320.

[5] Dantzig, G. B. [1963]. *Linear Programming and Extensions*, Princeton University Press.

[6] Hakopian, H. [1982]. "On Multivariate B-splines," in *SIAM Journal of Numerical Analysis* **19**., pp. 510–517.

[7] Höllig, Klaus [1981]. "A Remark on Multivariate B-splines," in *Journal of Approximation Theory* **33**, pp. 119–125.

[8] Höllig, Klaus [1982]. "Multivariate Splines," in *SIAM Journal of Numerical Analysis* **19**., pp. 1013–1031.

[9] Mangasarian, Olvi [1978]. "Linear Programming Lecture Notes."

[10] Micchelli, C. A. [1979]. "On a Numerically Efficient Method for Computing Multivariate B-splines," in *Multivariate Approximation Theory*, W. Schemp and K. Zeller, eds., ISNM 51, Birkhäuser, Basel, 1979.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 2744	2. GOVT ACCESSION NO. AD-A147386	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) The Stable Evaluation of Multivariate B-Splines		5. TYPE OF REPORT & PERIOD COVERED Summary Report - no specific reporting period
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Thomas A. Grandine		8. CONTRACT OR GRANT NUMBER(s) DAAG29-80-C-0041
9. PERFORMING ORGANIZATION NAME AND ADDRESS Mathematics Research Center, University of Wisconsin 610 Walnut Street Madison, Wisconsin 53706		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Work Unit Number 3 - Numerical Analysis and Scientific Computing
11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office P.O. Box 12211 Research Triangle Park, North Carolina 27709		12. REPORT DATE September 1984
		13. NUMBER OF PAGES 14
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

B-spline, simplex spline, multivariate, recurrence relation, linear programming, simplex method.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper gives a general method for the stable evaluation of multivariate B-splines. The problem of evaluation along mesh boundaries is discussed in detail. Several examples are presented to demonstrate the effectiveness of the method for arbitrary B-splines.

END

FILMED

12-84

DTIC